Week 7 Lab

For an example of how to do the exercises you can refer to my week 7 GitHub repo which is at:

https://github.com/oit-gaden/Web-Development-2019-Winter/tree/master/Examples/Week7

Don't forget to push your code to GitHub in a folder called week6 and your Docker images (you will have 2) to Docker Hub.

Preparation

1. Copy your week 6 folder to make a week 7 folder.
2. Go to **https://auth0.com/** and create an account. You can use your GitHub account to login or one of the other options if you want.

Exercise 1 - Secure your REST API

1. At **Auth0** create an API by selecting "APIs" from the dashboard.
    1. Click "Create API".
    2. Name your API "WebAPI".
    3. Give it a URL identifier (e.g. http://[your auth0 username]/webapi)
    4. Leave the signing algorithm as is (RS256).
2. In a shell/cmd window navigate to your week 7 webapi folder and run the following command to install a JWT token library for C#:

    dotnet add package System.IdentityModel.Tokens.Jwt

3. Follow the steps under the "Quick Start"
    1. For step one (Choose a JWT library) just run the following command in your webapi folder:

        dotnet add package System.IdentityModel.Tokens.Jwt

    2. Choose C# for step 2 and add the code shown to your **Startup.cs**. Just the one statement under the comments:

        // 1. Add Authentication Services

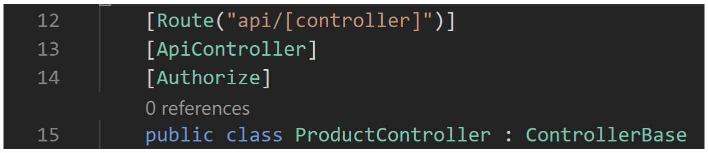
// 2. Enable authentication middleware

The strings shown should be in double quotes and not single quotes.

You'll need to add the following namespace to Startup.cs as well:

using Microsoft.AspNetCore.Authentication.JwtBearer;

4. Add the [Authorize] attribute to the student and person controllers as shown here for my products controller:



```
[Route("api/[controller]")]
[ApiController]
[Authorize]
0 references
public class ProductController : ControllerBase
```

5. Add the following namespace to both controllers as well:

   using Microsoft.AspNetCore.Authorization;

6. Use "Start without debugging" to start the webapi. Use Postman to try accessing your API for either student or person (e.g. http://localhost/5000/api/person).
   You should get back an HTTP status code of 401 for unauthorized. If you can't get VSCode to run your API then you can create a Docker container for the API
   and access it from Postman.

7. Use Postman to get a JWT token by doing an HTTP GET with the following URL:

   https://[your auth0 login name].auth0.com/oauth/token

   The request needs to be a **POST**.

   You need to add a request header of **Content-Type** with value of **application/json**.

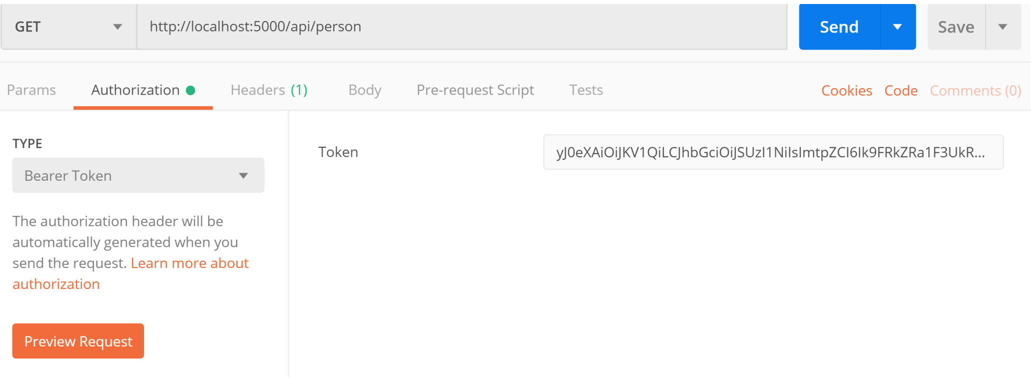
   The body of the request should be like this:

```
{
 "client_id": "CLIENT ID",
 "client_secret": "CLIENT SECRET",
 "audience": "http://[YOUR AUTH0 LOGIN NAME]/webapi",
 "grant_type": "client_credentials"
}
```

You get the CLIENT ID and the CLIENT SECRET by going to the Auth0 Applications section and finding the test application that matches your WebAPI.
For example: webapi (Test Application)

The JWT token will be the value of the "access_token" returned in the bottom window of Postman.

8. Use that token to add a authorization header to the HTTP GET request you created in step 6 above. Click on "Authorization", select "Bearer Token" from the dropdown,
put the token from step 7 in the "Token" field and resubmit the request. You should now see your data returned.



TYPE
Bearer Token

The authorization header will be automatically generated when you send the request. Learn more about authorization

Token yJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiIsImtpZCI6Ik9FRkZRa1F3UkR...

Exercise 2 - Add secure API access to your Web application frontend

1. In Auth0 create an application by selecting "Applications" from the dashboard.

    1. Click on "Create Application"r
    2. Name your application "WebApp"
    3. Select "Single Page Web Applications"
    4. Click "Create"
    5. On the next screen select "Vue"
    6. Under Auth0 Applications, click on the test application for your WebAPI, "Test Application (WebAPI). Go to the settings and
       set the following:

       "Allowed Callback URLs"
       to **http://localhost:8080/callback**

"Allowed Web Origins" to **http://localhost:8080**

2. Copy the "**auth**" folder in **src** and what it contains "**AuthService.js**" from my week 7 webapp example to your **webapp/src** folder.

3. Install the Auth0 SDK by running the following commands:

   npm install --save auth0-js
   npm install --save eventemitter3

4. Edit AuthService.js statement beginning with "auth0 = new auth0.WebAuth" as follows:
   1. Change domain to be your Auth0 WebApp domain (you'll find the domain on the Auth0 settings page for your WebApp).
   2. Change clientID to be your Auth0 WebApp clientID (you'll find the clientID on the Auth0 settings page for your WebApp).
   3. Change audience to be your Auth0 WebApi audience (you'll find the audience on the Auth0 APIs page along side of WebApi)

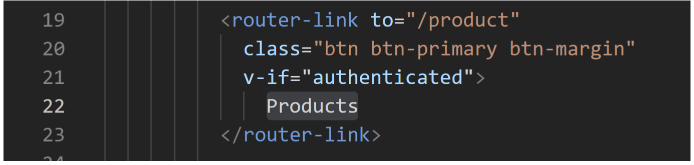
5. Install Bootstrap for styling the buttons added in the next step by running the following command in your webapp folder:

   npm install bootstrap

   Bootstrap was not covered in class but a good CSS framework to become familar with.

6. Replace your App.vue component with the one in my week 7 webapp folder.

   Edit the following code to change /product to /person and "Products" to "Persons":
   person



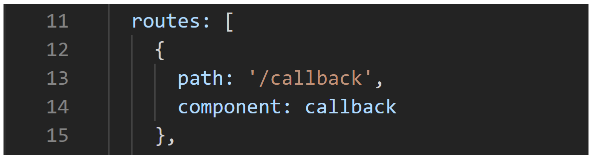
```
<router-link to="/product"
  class="btn btn-primary btn-margin"
  v-if="authenticated">
    Products
</router-link>
```

   Add another "router-link" like that one for the "Students" entity.

7. Copy the Callback.vue component from my week 7 webapp components folder.

8. In the router/index.js file:

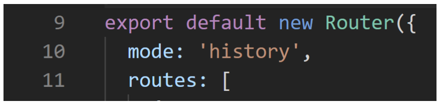
   a. Add a route for the Callback component like this:



```
routes: [
  {
    path: '/callback',
    component: callback
  },
```

   b. Remove any route for "/".

   c. Set the routing mode to use browser history like this:



```
export default new Router({
  mode: 'history',
  routes: [
```

d. Add an import for the Callback component.

9. Run npm start and navigate your browser to the URL shown (http://localhost:8080/

    1. You should see the App.vue component displayed with the "Login" button. Click it and you should be
    redirected Auth0 for logging in or signing up.

    2. Signup a user (you can find the user you signup under the Users section at Auth0). Notice the password
    policy display as you type in a psssword.

    3. It will then ask you for authorization which you should approve.

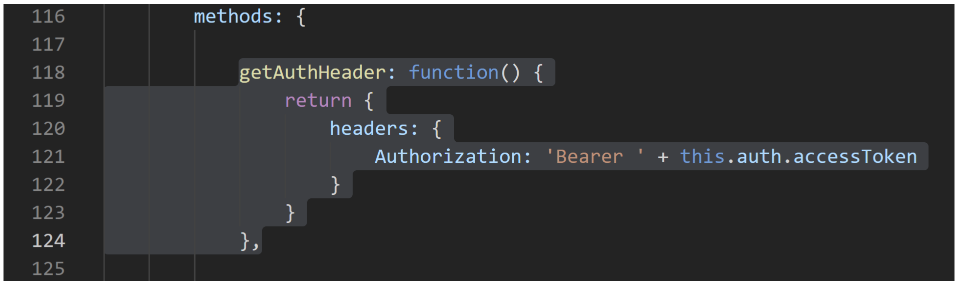
    4. You shoud not be taken to your "home page".

    5. If you attempt to access "Persons" it will fail as you now need to add the code to pass the token to the API for
    authentication.

10. Add the following line after the component name field in your "Persons.vue" component: (see my Products.vue component)

    props: ['auth'],

11. Add the getAuthToken method to your Persons.vue component as shown here: (see my Products.vue component)

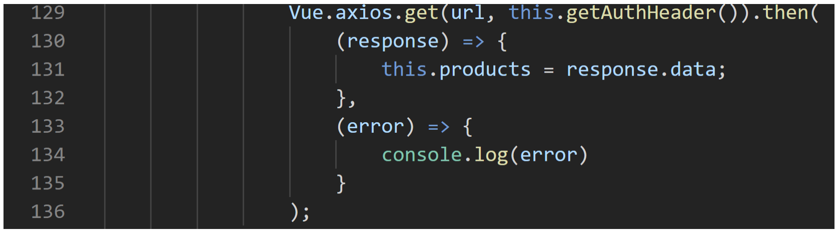


```
        methods: {

            getAuthHeader: function() {
                return {
                    headers: {
                        Authorization: 'Bearer ' + this.auth.accessToken
                    }
                }
            },

```

12. Add a call to getAuthToken() as the last parameter to all of your Vue.axios REST calls as show here:



```
                Vue.axios.get(url, this.getAuthHeader()).then(
                    (response) => {
                        this.products = response.data;
                    },
                    (error) => {
                        console.log(error)
                    }
                );
```

13. You should now be able to run the application and see your data as before. Now the data is being retrieved from the API
with authorization.

14. Push your Docker images to Docker Hub and push your code to GitHub.